IMPLEMENTING A GENERIC FRAMEWORK FOR A WEB-BASED PEDAGOGICAL AGENT

[1.]Smith, T., Affleck, G., Lees, B. and Branki, C.

Computing and Information Systems
University of Paisley
Scotland, UK

[1.]Email: smit-ci2@paisley.ac.uk

Abstract

The implementation of a Web-Based Pedagogical Agent framework to the domain of a programming course support application is detailed. Research describing the need for a Pedagogical Agent is presented, the system framework and technical issues of implementation are described, followed by a discussion of necessary changes to the original model.

Keywords

Pedagogical Agent, Interface Agent, Web-Based Education, Active Learning, Novice Programmers, Multi-Pedagogical-Agent System.

1. The need for a pedagogical agent

The World Wide Web has been heralded as an optimal medium for provision of educational materials for both distance and local courses. However, while Web-based courseware provides a more flexible, individually-paced approach than traditional classroom based learning materials, online course support systems can still be impersonal, confusing and thus de-motivating to many students. Moreover, in both computer-based and traditional learning environments, students often experience frustration as a result of a perceived need to keep up with other students, reluctance to ask basic questions, delays in constructive feedback about their work, a lack of connection with peers struggling with similar problems, and uncertainty of the best methods of structuring their own learning process.

This paper describes a framework for the practical application of interface agent technology as a means of solving some persistent problems in educational environments. When integrated into a more conventional web-based course support system, a pedagogical interface agent can provide an educational environment which is not only flexible and interactive, but also personalized, emotionally responsive, and adaptive to a student's unique needs for structure, pacing and feedback.

2. Current capabilities of pedagogical agents

Pedagogical Agents are a relatively new branch of interface agents which are designed to assist in the educational process of humans in a variety of domains. Interface Agents are in turn a branch of the field of software agents which exist as the primary point of contact between a human and a computer. Their style of implementation can range as wide as currently existing graphical user interfaces or a

voice over a telephone (Moran, et al 1997) to a life-size, full-body 3D image in a virtual room (Maes 1995). The type of interface agent most frequently implemented to date however is a 2D miniature full-body animated cartoon figure which appears to float over a graphical user interface windows environment or web page.

Pedagogical Agent Technology has already illustrated many capabilities such as the coordination of speech and actions (Elliott, 1997), the monitoring of student actions (Johnson, 1997), the integration of spoken language input (Ball, 1996), and the application of constructivist learning theories (Lester, 1997a). These agents can now also adapt their behaviors to both the environment and the student, offer opportunistic instruction or hints, and can support collaborative learning. (Johnson, 1999).

Recent studies have produced results which indicate a variety of advantages in implementing interface agents as part of educational applications. In her study of the effects of the personification of agents (Koda & Maes, 1999), Tomoko Koda found that in an interactive application, an embodied interface agent (as opposed to a disembodied dialogue box) was more engaging to the user. The creators of WebPersona (Andre, et al, 1998) discovered that their subjects rated learning tasks presented by the Persona agent as less difficult than the presentations viewed without such an animated, graphically depicted interface agent. Initial studies of a plant biology tutoring agent by (Lester, et al, 1997c) revealed that lifelike, personable interface agents were perceived by students as being very 'helpful, credible and entertaining' and that agents which offer a range of levels of advice can increase learning performance. A study by (Klein, 1999) has shown that an interface agent designed to support users in managing and recovering from negative emotions can encourage user persistence at a difficult task.

3. Proposed framework for a generic pedagogical agent

To incorporate the many benefits of Interface and Pedagogical Agents in educational applications, a framework for a generic pedagogical agent is proposed. The idea being that rather than creating a pedagogical agent from scratch at each instance, the basic components of a generic framework could be more easily applied to variety of specific education domains. Of course this raises the issue of level of granularity - the more specifically the generic framework is defined, the easier it is to implement. However, A generic framework that is too specific may not be general enough to apply to many domains. This issue can only be resolved by an experimental process of design and implementation such as the one described in this paper.
It is suggested that the generic pedagogical agent consist of 6 modules as follows:

The User Preference Module - contains a database of individual student's preferences (commonly known as a 'user model') regarding the form of the agent (embodied animation or dialogue box), the appearance and functional details of the system and preferred methods of working and is updated and maintained by the agent. This module is primarily responsible for personalization ability of the agent.

The Behavior Module --contains the agent's scripted actions such as gesturing and moving across the screen, as well as the idle time actions randomizer (which rolls through a series of behaviors related to the current actions of the student), and a library of responses based on student actions and questions.

The Decision Making Module - contains pacing decisions, such as when to allow the student to the next level, when to offer suggestions, hints, presentations, example codes and links to other sources of human support as well as interpretation decisions based on the student's current work which allows the agent to act as facilitator.

The Student Progress Module - contains the past record of a student's performance and information regarding the student's current work which, when combined with the functionality of the decision-making module, allows the agent to act as a tutor.

The Emotional Support Module - contains the active listening guidelines, a case base of possible solutions to common problems, and reasoning ability to generate possible new solutions based on past solutions thus enabling the agent to act as advisor.

The Communication Module - contains the text and speech interpreter and the text-to-speech engine enabling the agent to communicate with a student via speech, mouse or keyboard. The communication module, in cooperation with the behavior and decision-making modules forms the basis of the agent's interactivity and adaptive capabilities.

4. The three hats of a generic pedagogical agent

The Generic Pedagogical Agent would perform three main functions within a course support application. As a facilitator, it helps direct the student through the learning environment in the manner best suited to each individual. As a tutor, it promotes of active learning by offering facilities and exercises which help the student learn to teach her- or himself. As an advisor, it displays some emotional responsiveness and problem solving capability.

Facilitator

While in the role of the facilitator, the generic pedagogical agent would structure the course of learning according to individual student progress and preference. It would act as a interactive gateway to the many features of the course support system such as instructional presentations, paced exercises, examples, demonstrations, a database of common student misconceptions, contact to peers who are struggling with or have solved similar problems, contact to tutors and lecturer as well as lab and lecture materials. As facilitator, the generic pedagogical agent should be both reactive and proactive. It will be able to respond to direct student queries via voice, mouse or keyboard as well as offer suggestions to resources during the supervision of interactive exercises as a tutor.

Tutor

As a tutor, the generic pedagogical agent would encourage active learning through a series of interactive and individually paced exercises. These exercises must be carefully created so that it is possible for the agent to accurately assess when they have been correctly completed by a student. The agent would guide a student through practice exercises using a series of hints, suggestions and demonstrations. Depending on the level of scaffolding required by a particular student, the agent would provide

more or less detailed or general guidance, until the student is able to perform satisfactorily in a unit of work. As a tutor, the agent will be programmed to speak (in voice or word balloons) with concrete, non-technical, language as much as possible. The agent responds with a range of enthusiastic phrases and motions when the student is successful at a given task and sympathetic phrases and motions when the student is unsuccessful. These mechanisms assist in creating a personable and entertaining side to the pedagogical agent.

Advisor

As an advisor, the generic pedagogical agent displays a degree of emotional intelligence. Using a combination of reflective or active listening and offering suggestions as to which resources may help solve the mentioned problem, the agent may alleviate the frustration which can lead to de-motivation. A case-based reasoning system will be used to generate new solutions to new problems based on a repository of typical solutions to common problems. Initially this feature will be a student-chosen option to be selected whenever the student feels they need to let off steam. Later implementations may explore methods of perceiving user emotional state.

5. Incorporation of a generic pedagogical agent into a programming course support application

Why Programming?

Learning to program is notoriously difficult process in part due to the proliferation of abstract concepts, but perhaps also due to the dearth of programming courses which are truly designed for absolute beginners. Due to the above reasons, a study was performed by the authors which illuminated a series of commonly experienced problems by students participation in a beginning programming. (Affleck & Smith, 1999). In this study it was found that novice programming students report three primary difficulties in learning to program. Firstly, students reveal problems in relating the new concepts being presented to the real world they already know. Secondly, students describe problems in visualizing abstract programming concepts. Thirdly, students report major difficulties in moving from being able to passively understand a computer program to being able to actively write a program.

Features of the existing system

The existing programming course support application, into which the pedagogical agent will be incorporated, already has many interesting features. It is integrated into a visual basic environment by means of extra buttons on the taskbar. From these buttons a student may access a series of powerpoint presentations which illustrate sample code and programming techniques. A student may also access sets of interactive exercises, a database of common student misconceptions, links to lecture and lab materials as well as contact information for tutors, lecturer and other students.

Advantages of incorporating a pedagogical agent into the system

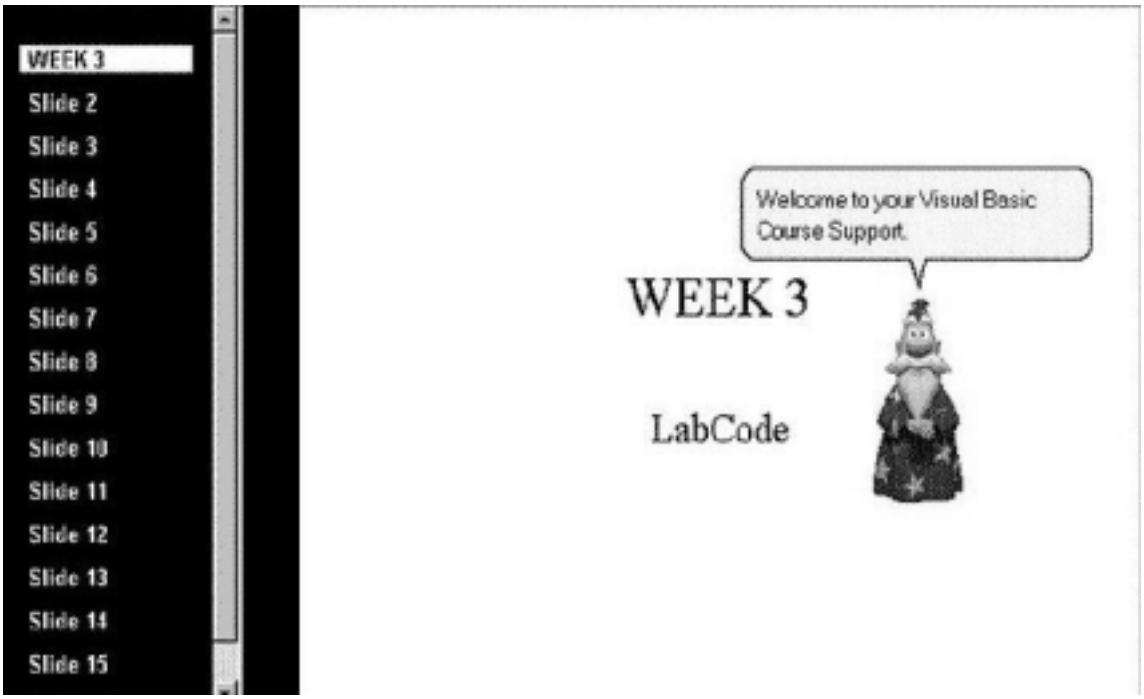
The Pedagogical Interface Agent creates a central focus point to the course support system, playing the role of a personal assistant or tutor guiding the student in the development of an active learning style. The agent acts as a gatekeeper, offering flexible access to the system including on-line help facilities and reference materials as well as synchronous and asynchronous access to human support. As a personable, natural language speaking guide through the course support system, the agent enables ease of use especially for novice users. It provides a simpler interface for a student so that there is no effort wasted in learning how to use the learning tool. As an animated interface cartoon-type character with a range of behaviors and reactions, the agent is intended to be entertaining as well as educational in order to engage and maintain student interest while promoting motivation. The Agent is interactive, providing immediate constructive feedback to exercises and questions by a personalized agent which is able to adapt to each individual learner's needs through the utilization of a student preference and progress profile. The agent possesses a concrete tutoring style which relates new abstract concepts to portions of the real world with which students are already familiar. The Agent promotes active learning by pacing the work of the student, directing the student to sources of additional assistance (in the form of examples, hints, short tutorials, links to other students who solved similar problems), and encouraging the student to move on to the next level only when they are fully capable.

Technical issues

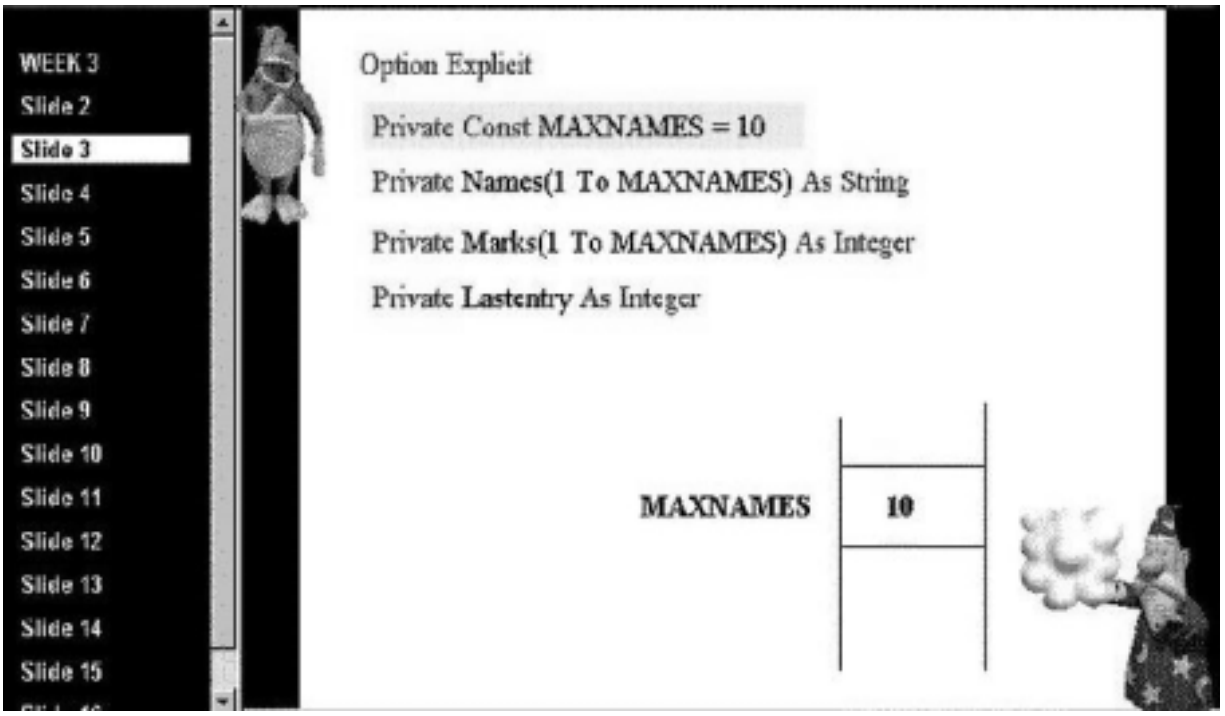
The Agent is embedded into an already existing course support system using in the first instance an adaptation of the Microsoft Agent Technology. The initial implementation language is VBscript which will later to be converted to Java for platform independence. Microsoft Agent was chosen from a list of seven recently available interface agent development tools because it offers a wide range of useful features. There is a pre-existing library of animations as well as text-to-speech and vocal language interaction capabilities. Since the authors do not have the benefit of a team of animators, graphic artists and sound engineers, these built-in functions of MS Agent make it possible to focus on the (rule-based) decision-making, emotional reasoning, problem-solving, monitoring, adaptive and interactive capabilities of the Agent design as well as on the application of learning theories to the system as a whole. A database system will be used to keep track of students who have already solved particular problems successfully so that the agent will be able to match up student who are struggling with a particular area with those who can be considered 'peer experts'. To assist the agent in recognizing the student's stream of thought while programming, students will be encouraged to follow a set procedure of initially describing problems in high-level and then subsequently lower-level algorithms. Programming tasks will be broken down into code modules relating to these algorithms. The Agent system is designed to be a de-centralized client-side application which will be based as a floating image on the programming environment thereby acting as a personalized gateway to system features.

New hats — adapting the model

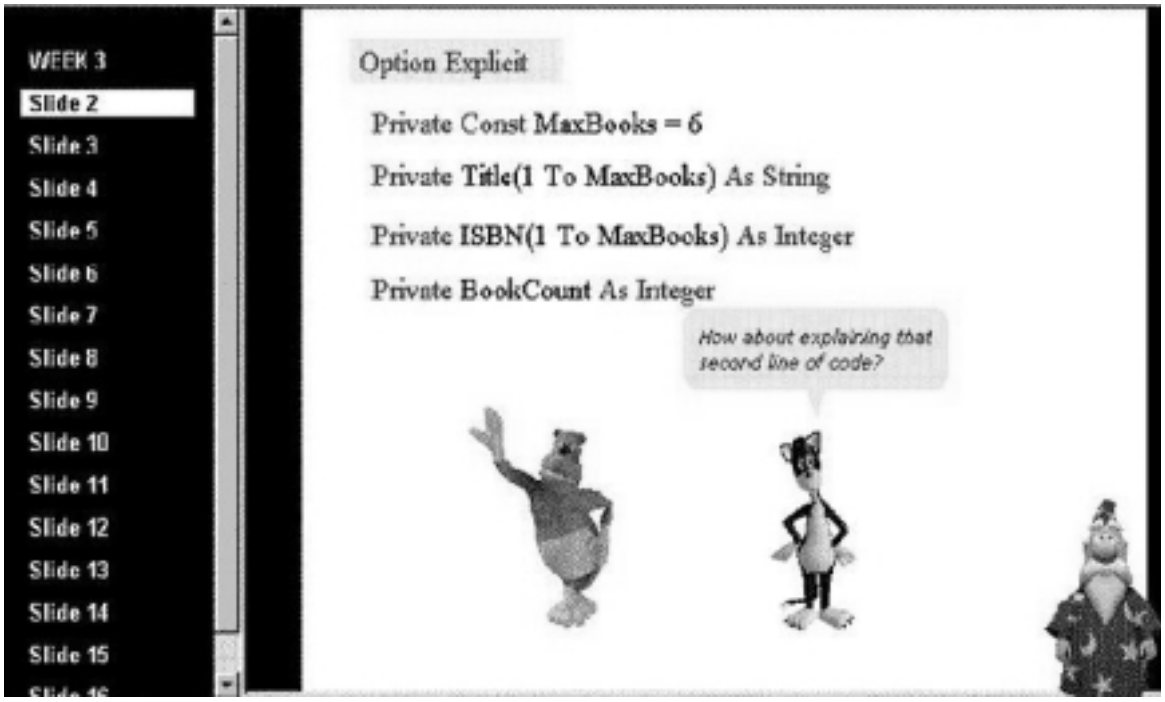
It transpires that it is not actually possible for a Microsoft Agent which has been incorporated into a web page as a presentation agent to also carry out system commands. For this reason it was necessary to create another agent as a standalone executable file which is accessible from a student's task bar and has the capability to carry out the necessary system commands. This meant a reformation of the original generic model to include a set of four interacting interface agents – a Facilitator , a Tutor, a Psuedo-Student and a Assistant/Advisor. These four agents have been implemented into the programming course support system as follows:



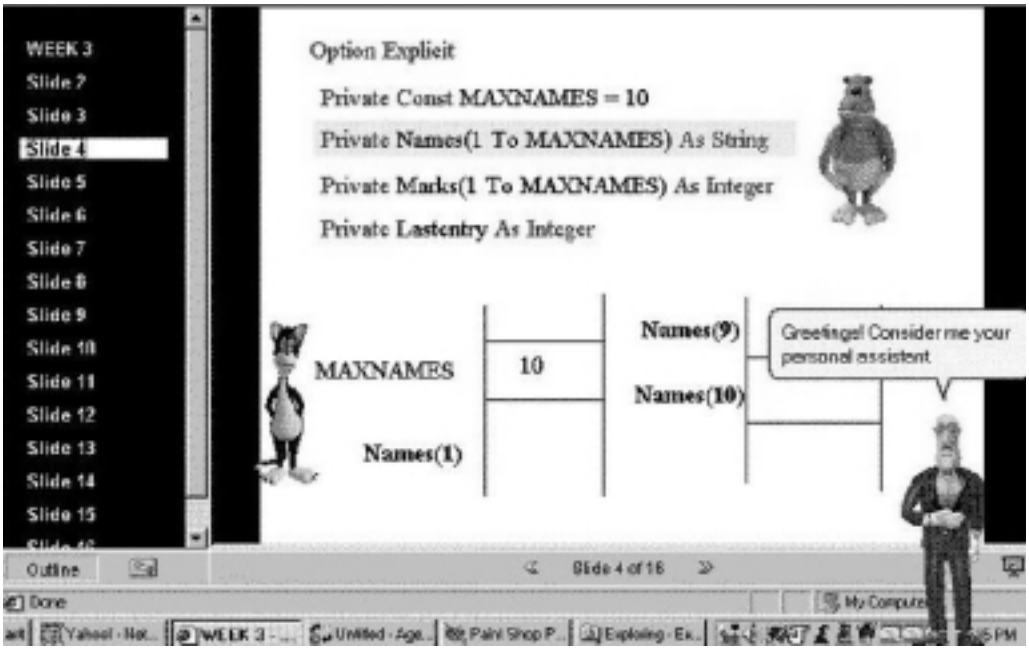
The Facilitator Agent acts as an introducer to the application features such as the presentation files as illustrated above. Since it is strictly a presentation agent, it is not interactive, but instead serves the purpose of guiding students through the system and directing them to various features, particularly the interactive, executable Assistant Agent.



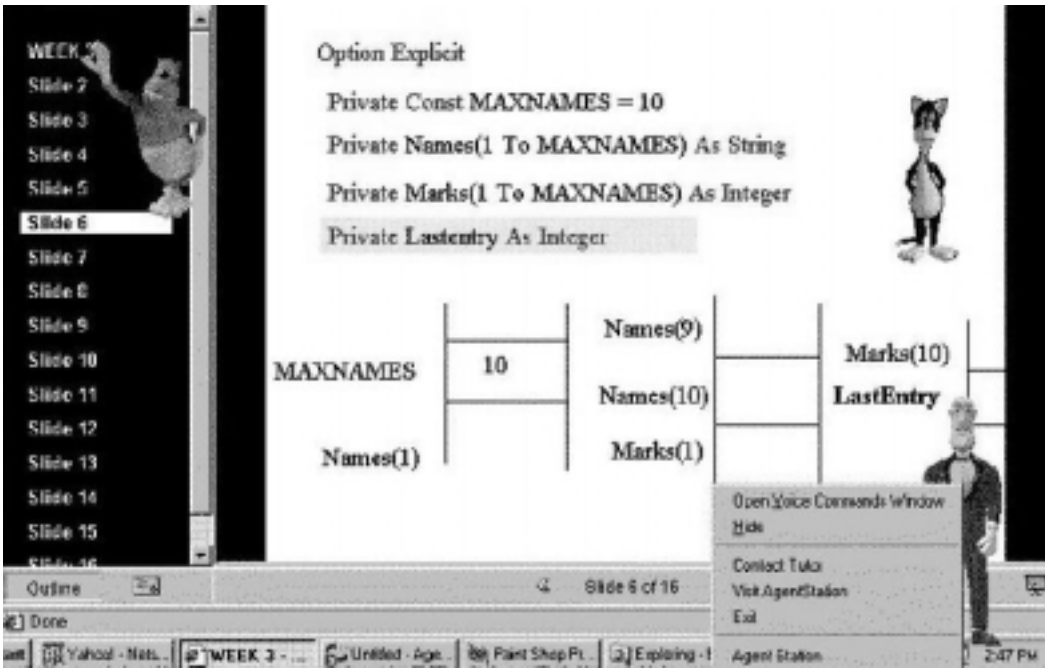
The Tutor Agent has a dual implementation as a presentation agent (seen above describing a line of code) and as a stand alone .exe file capable of providing interactive exercises. Initial plans for exercises include syntax recall, code segment building, algorithm building, writing programs from pseudo-code and debugging exercises. Throughout the various stages of exercises, the agent will be able to provide immediate feedback and suggestions about a student's learning process.



The Psuedo - Student Agent (seen tapping foot impatiently above) performs a dual presentational role. It not only provides occasional cheekiness and comic relief, it briefly summarizes the content of key slides, and asks typical beginning programming student questions.



The Assistant/Advisor Agent introduces itself to the student as a personal assistant. Along with the Tutor .exe agent, it is fully interactive in a variety of modes as describes below.



The student may right click on the agent to allow a selection menu to appear as seen above. Then the student has the option of either selecting a command or speaking key words aloud. The student may also choose a dialog box option and communicate with the agent in natural typed or spoken language. As with all current voice activation systems, some training may be necessary. The agent will respond to the student in either typed word balloon fashion or text-to-speech generation, or by carrying out the requested command.

6. Related research

Many pedagogical interface agents are being or have already been applied in educational applications relating to plant biology (Lester, et al, 1997c) medicine case diagnosis Johnson & Shaw, 1997), Internet packet routing (Lester, et al, 1997b), technical information presentation (Andre, et al, 1998), and physical procedure task sequencing (Elliott & Brzezinski, 1998).

The Pedagogical Agent offers a novel approach in comparison with these existing applications in a number of aspects. Firstly, the Pedagogical Agent is aimed at the notoriously difficult domain of adult novice programmers. It incorporates constructivist theory by incorporating a task-based model which proceeds in very small stages according to each student's Zone of Proximal Development (ZPD) (Van der Beer & Valsiner, 1993). An important element of incorporating the ZPD element of constructivist theory is proper pacing. The Pedagogical Agent maintains a student progress model and guides the student through several levels of programming expertise, but encourages the student to remain at a relevant level, continuing with the interactive exercises, demonstrations and presentations until the student is fully ready to move on to the next level. Finally, the proposed Pedagogical Agent will offer an emotional support element.

7. Conclusions and further research

In short, the ideas for the Pedagogical Agent grew out of a study of actual student needs and will continue to have a close relationship with the requirements of the student population in a series of prototype tests. Implementation of an initial prototype is well underway for use as a testbed to determine the following:

1. Do students learn (more quickly, more easily, more thoroughly) with or without the Pedagogical Agent?
2. Are students more likely to ask basic questions of the Pedagogical Agent?
3. Are students more likely to seek out additional sources of support if the Pedagogical Agent describes the support options and provides an immediate contact connection?
4. Do students find the emotional empathy feature useful? Does it increase motivation to continue though difficult areas?
5. Do students find system more enjoyable/entertaining with the Pedagogical Agent?
6. Do students experience less frustration or confusion with the Pedagogical Agent?
7. Is motivation positively affected? Do students spend more time on the course support with the Learning Assistant Agent? Do they indicate a stronger desire to continue or a higher chance of success? Does the system increase self-confidence?

Does the interaction of various agent roles/hats increase enjoyment/ comprehension of the presented material?

By engaging the student as a personalized, adaptive approachable interface through which the student can access properly-paced tutorials, examples, demonstrations, line-by-line code descriptions as well as peer experts, the agent system promotes the critical move from passive to active learning - essentially helping students learn to teach themselves. Additionally, in order to help combat the eventual de-motivation of students who are unable or unwilling to air the frustrations commonly experienced while learning a highly abstract and complex subject, an emotional support/problem solving module of the Pedagogical Agent will be available as part of the programming course support system.

References

Affleck G, and Smith T. 1999. Identifying a Need for Web-Based Course Support, *ED-MEDIA Proceedings, 1999*.

Andre, E, Rist, T, Mueller, J. 1998. WebPersona: a lifelike presentation agent for the World-Wide Web, *Knowledge Based Systems*, 2 (1), 25-35.

Ball, G, et al. 1996. Lifelike Computer Characters: the Persona Project at Microsoft Research, Microsoft Research Web page,

http://www.research.microsoft.com/ui/ __http://www.research.microsoft.com/ui/_

Elliott, C & Brzezinski, J. 1998. Autonomous Agents as Synthetic Characters, *AI Magazine*, 19 (2) 13-30.

Elliott, C. 1997. Integrating Affective Computing into Animated Tutoring Agents, http://www.depaul.edu/~elliott/papers/ijcai97/ij.html

Elliott, J. 1997. Coordinating Speech and Actions for Animated Pedagogical Agents, Thesis at North Carolina State, http://www.csc.ncsu.edu/degrees/undergrad/Reports/jlelliot/thesis97.html

Johnson, WL, 1999. Pedagogical Agents, http://www.isi.edu/isd/carte/ped_agents/pedagogical_agents.html

Johnson, W L & Shaw, E. 1997. Using Agents to Overcome Deficiencies in Web-Based Courseware, *IJCAI-97 Animated Interface Agents Workshop 1997*.

Klein, J T. 1999. Computer Response to User Frustration, MSc Thesis at Massachusetts Institute of Technology.

Koda, T and Maes, P. 1996. Agents with Faces: The Effects of Personification of Agents, *Proceedings of HCI'96*, London, 1996.

Lester, J C, et al. 1997(a). The Pedagogical Design Studio: Exploiting Artifact-Based Task Models for Constructivist Learning, *IUI Conference Proceedings 1997*.

Lester, J C, et al. 1997(b). Cosmo: A Life-Like Animated Pedagogical Agent with Deictic Believability, *IJCAI-97 Animated Interface Agents Workshop 1997*.

Lester, J C, et al. 1997(c) The Persona Effect: Affective Impact of Animated Pedagogical Agents, *CHI Conference Proceedings 1997*.

Moran, D B, et al. 1997. Multimodal User Interfaces in the Open Agent Architecture, *IUI Conference Proceedings 1997*.

Maes, P. 1995. Artificial Life meets Entertainment: Interacting with Lifelike Autonomous Agents." *Special Issue on New Horizons of Commercial and Industrial AI*, 38 (11), 108-114, Communications of the ACM, ACM Press.

Van der Beer, R & Valsiner, J. 1993. *Understanding Vygotsky*, Blackwell Publishers, Cambridge, Massachusetts.